HW5 Documentation

Itamar Barron – 208981159
Roy Frumkis – 312472129

Introduction

The project is a program designed to work with a FAT12 file system. It provides functionalities such as listing the contents of the root directory and copying files from the root directory to the destination directory.

The program is built from a collection of functions, under the main files, that enables navigation and manipulation of the FAT12 disk image. These functions perform tasks such as reading the boot sector, extracting the content from files based on their entry points in the file allocation table (FAT) and checking the existence of files in the root directory. By understanding these functions and their implementation, we can understand the FAT12 file system and its structure.

Course of action

The program starts by checking the command-line arguments, It verifies the number of arguments and ensures that the second argument is either "dir" or "cp" as asked in the guidelines.

If the command is "dir":

- The program opens the file specified in the first argument.
- It reads the boot sector of the FAT12 file system from the file.
- The program calls the "*print_root_dir*" function to display the contents of the root directory with *"seekset"* and making few tests on the *dir.entry*.
- Finally, the program closes the file, parse the data into the desired format and printing.

If the command is "cp":

- The program opens the file specified in the first argument.
- It reads the boot sector of the FAT12 file system from the file.
- The program checks if the source file specified in the third argument exists in the root directory.
- If the file exists:
    - The program converts the filename to uppercase.
    - It retrieves the starting entry of the file from the FAT12 file system.

    - The program extracts the file from the FAT12 file system and saves it with the destination filename specified in the fourth argument.

- Finally, the program closes the file.

Different files:

1. **`main.c`:** This is the source file written in the C programming language that contains the main function for your program. It provides the execution instructions to interact with a FAT12 file system, providing functionalities such as printing the root directory, fetching file information, navigating through FAT entries, and extracting files.

2. **`main.h`:** This is the header file associated with the `main.c` source file. It declares the function prototypes used in `main.c` and also defines a couple of constants, providing a centralized place to update these values if needed.

3. **`msdos_fs.h`:** This header file contains definitions, constants, and data structures relevant to the MS-DOS file system, specifically targeting the FAT (File Allocation Table) system. It includes low-level details about boot sectors, directory entries, file attributes, etc., used to interact with MS-DOS formatted file systems.

4. **`Makefile`** : The is a build automation tool that compiles your C source file `main.c` into an executable file named `hw5`. It also includes a `clean` target that removes the compiled executable, allowing you to rebuild the project from scratch.

Main Fucntions:

The following functions are the core FAT12 functions the uses the FAT12 file system logic. All function are described in main.c file.

1. **`print_root_dir(FILE* img_file, struct fat_boot_sector* boot_sector)`:** This function is responsible for presenting the content of the FAT12 root directory in a structured format. It takes an image file and a boot sector as arguments, reads the root directory entries in the image file, which are structured according to the `msdos_dir_entry` struct. Each entry contains a file name, attributes, time and date stamps, and the cluster number where the file starts. This function iterates over these entries, converts the non-human-readable data into a user-friendly format, and then prints it.

2. **`get_file_info(FILE* img_file, struct fat_boot_sector* boot_sector, char* file_name , __le16* start_entry , __le32* file_size)`:** This function is designed to fetch detailed information about a specific file located within the FAT12 root directory. It scans the root directory (structured by `msdos_dir_entry`) for an entry matching the provided file name. When it locates the file, it fetches the start cluster (indicating where the file data starts) and file size from the directory entry, which are then stored in the `start_entry` and `file_size` parameters.

3. **`get_next_entry(FILE* img_file, struct fat_boot_sector* boot_sector, __le16 current_entry)`:** This function helps navigate through the FAT12 File Allocation Table (FAT).

FAT is a linked list of entries where each entry corresponds to a cluster in the data region, indicating the next cluster in the file or signaling the end of the file. Given the current cluster number, this function fetches the next cluster number from the FAT, facilitating file reading across multiple clusters.

4. **`extract_file_from_fat12(FILE* img_file, struct fat_boot_sector* boot_sector, __le16 start_entry , char * name , __le32 file_size)`:** This function aims to extract a specific file from a FAT12 image. By using the start cluster of the file and its size, the function follows the linked clusters in the FAT, reading data from each cluster until the file's entire content is fetched. The extracted data is then written to a new file in the host filesystem with the provided name.

5. **`main(int argc, char* argv[])`:** As the entry point of the program, this function takes command-line arguments and interprets them to determine the operations to perform. If the 'dir' command is detected, it reads the boot sector of the FAT12 image file and calls `print_root_dir` to print the root directory content. If the 'cp' command is identified, the function not only checks for the file's existence in the root directory but also retrieves the file's information using `get_file_info` before finally extracting the file to the host filesystem with `extract_file_from_fat12`.

Helper Fucntions:

The following function are added helper functions that don’t have any direct relation to the FAT12 file system. All function are described in the main.c file.

1. **`time_to_string(__le16 ctime , char* time_str)`:** This function converts the creation time of a file (stored in a FAT-specific format) to a human-readable string.

2. **`date_to_string(__le16 cdate , char* date_str)`:** Similar to `time_to_string`, but converts the creation date instead.

3. **`fat_name_to_normal_name(char* FAT_name, char* normal_name)`:** This function converts a filename from the format used in the FAT12 filesystem to a normal string.

4. **`format_number(__le32 num, char *str)`:** This function converts a 32-bit little-endian number to a string, formatting it with commas as thousands separators.

5. **`check_if_file_in_root_dir(FILE* img_file, struct fat_boot_sector* boot_sector , char *file_name)`:** This function checks if a specific file exists in the root directory of the FAT12 image.

6. **`str_to_upper_case(char* str)`:** This function converts a string to uppercase. This is needed because filenames in FAT12 are stored in uppercase.

Summary

This document showcasing the program we wrote that works on the FAT12 File System. The program is specifically designed to interact with FAT12 disk files, file extract, directory listing and file copying.

By working on this project, we gained valuable experience in dealing with the FAT 12 file system. We learned how to extract information about files, list directories, and copy files. During writing the program, we also used operations to read and write files, manipulated strings and efficiently managed the FAT table. This hands-on experience helped us better understand the concepts we learned in class.